In the Claims:

Please amend claims 31-35 and 37 as indicated below.

1. (Previously presented) A system, comprising:

a plurality of application servers, wherein each of the plurality of application servers is configured to access session data, wherein the session data represents the state of a client session for a client; and

a distributed store comprising a primary state of the session data configured for access by the plurality of application servers, wherein the distributed store is configured to provide locked access to the primary state to a process executing within one of the plurality of application servers, wherein, while the primary state is locked for the process, other processes cannot access the primary state;

wherein in providing locked access to the primary state to a process executing within one of the plurality of application servers, the distributed store is configured to send a lock token to the process, wherein only processes that have received a lock token can access the primary state.

2. (Previously presented) The system as recited in claim 1, wherein, after the process has completed a current access of the primary state, the process is configured to hold locked access until after receiving a request to release the locked access.

3. (Original) The system as recited in claim 2, wherein the distributed store is configured to request the process to release the locked access, wherein the process is configured to release the locked access in response to said request.

4. (Original) The system as recited in claim 1, wherein the process is configured

to release the locked access when the process no longer requires the locked access to the primary state.

5. (Original) The system as recited in claim 1, wherein the process is configured to provide locked access to at least a portion of the primary state to a thread executing within the process, wherein, while the at least a portion of the primary state is locked for the thread, other threads executing within the process cannot access the at least a portion of the primary state.

6. (Original) The system as recited in claim 1, wherein the distributed store is configured to grant the locked access to the process executing in one of the application servers in response to a request for locked access from that process.

7. (Original) The system as recited in claim 1, wherein while the process holds the locked access, the distributed store is configured to buffer one or more requests for locked access from one or more other processes executing within one or more of the plurality of application servers.

8. (Original) The system as recited in claim 7, wherein, if the process releases the locked access to the primary state, the distributed store is configured to provide locked access to one of the other processes in response to the other process's buffered request.

9. (Original) The system as recited in claim 1, wherein another process executing within one of the plurality of application servers is configured to request locked access to the primary state from the distributed store, and wherein if no process currently holds locked access to the primary state, the distributed store is configured to provide locked access to the primary state to the other process.

10. (Previously presented) A system, comprising:

a plurality of application servers, wherein each of the plurality of application servers is configured to access session data, wherein the session data represents the state of a client session for a client; and

a distributed store comprising a primary state of the session data configured for access by the plurality of application servers, wherein the distributed store is configured to provide locked access to the primary state to a process executing within one of the plurality of application servers, wherein, while the primary state is locked for the process, other processes cannot access the primary state;

wherein the process is configured to provide locked access to portions of the primary state to one or more threads executing within the process, wherein, while a portion of the primary state is locked for one of the threads, other threads executing within the process cannot access the particular portion of the primary state;

wherein the distributed store is configured to request the process to release the locked access, wherein the process is configured to release the locked access in response to said request.

11. (Previously presented) The system as recited in claim 10, wherein, after the process has completed a current access of the primary state, the process is configured to hold locked access until after receiving a request to release the locked access.

12. (Canceled)

13. (Original) The system as recited in claim 10, wherein the distributed store is configured to grant the locked access to the process executing in one of the application servers in response to a request for locked access from that process.

14. (Previously presented) The system as recited in claim 10, wherein the distributed store is configured to buffer one or more requests for locked access from one or more other processes executing within one or more of the plurality of application servers.

15. (Original) The system as recited in claim 14, wherein, if the process releases the locked access to the primary state, the distributed store is configured to provide locked access to one of the other processes in response to the other process's buffered request.

16. (Previously presented) The system as recited in claim 10, wherein another process executing within one of the plurality of application servers is configured to request locked access to the primary state from the distributed store, and wherein if no process currently holds locked access to the primary state, the distributed store is configured to provide locked access to the primary state to the other process.

17. (Previously presented) A system, comprising:

a plurality of application servers, wherein each of the plurality of application servers is configured to access session data, wherein the session data represents the state of a client session for a client;

a distributed store comprising a primary state of the session data configured for access by the plurality of application servers; and

means for providing locked access to the primary state to a process executing within one of the plurality of application servers, wherein, while the primary state is locked for the process, other processes cannot access the primary state;

wherein said means for providing locked access comprises means for sending the

processing a lock token, wherein only processes that have received a lock token can access the primary state.

18. (Original) The system as recited in claim 17, further comprising means for providing locked access to at least a portion of the primary state to a thread executing within the process, wherein, while the at least a portion of the primary state is locked for the thread, other threads executing within the process cannot access the at least a portion of the primary state.

19. (Original) The system as recited in claim 17, further comprising means for releasing the lock when the process no longer requires locked access to the primary state.

20. (Previously presented) A method, comprising:

a process executing within one of a plurality of application servers requesting locked access to a primary state of session data stored by a distributed store; and

granting a lock to the process requesting locked access, wherein, while the process holds the lock, other processes cannot access the primary state in the distributed store;

wherein said granting comprises sending a lock token to the process, wherein only processes that have received a lock token can access the primary state.

21. (Original) The method as recited in claim 20, further comprising the process holding the lock granting a thread-level lock to a portion of the primary state to a thread running within the process, wherein, while the thread holds the thread-level lock, other threads cannot access the portion of the primary state.

22. (Original) The method as recited in claim 20, further comprising:

requesting the process release the locked access; and

the process releasing the locked access in response to said request.

23. (Original) The method as recited in claim 20, further comprising the process releasing the lock when the process no longer requires locked access to the primary state.

24. (Original) The method as recited in claim 20, further comprising buffering one or more requests for locked access from one or more other processes executing within one or more of the plurality of application servers.

25. (Original) The method as recited in claim 24, further comprising, if the process releases the locked access to the primary state, the distributed store granting locked access to one of the other processes in response to the other process's buffered request.

26. (Previously presented) A method, comprising:

a process executing within one of a plurality of application servers requesting locked access to a primary state of session data stored by a distributed store;

granting a lock to the process requesting locked access, wherein, while the process holds the lock, other processes cannot access the primary state in the distributed store;

the process holding the lock granting a thread-level lock to a portion of the primary state to a thread running within the process, wherein, while the thread holds the thread-level lock, other threads cannot access the portion of the primary state;

requesting the process release the locked access; and

the process releasing the locked access in response to said request.

27. (Canceled)

28. (Original) The method as recited in claim 26, further comprising the process releasing the lock when the process no longer requires locked access to the primary state.

29. (Original) The method as recited in claim 26, further comprising buffering one or more requests for locked access from one or more other processes executing within one or more of the plurality of application servers.

30. (Original) The method as recited in claim 29, further comprising, if the process releases the locked access to the primary state, the distributed store granting locked access to one of the other processes in response to the other process's buffered request.

31. (Currently amended) A computer-accessible storage medium, comprising software instructions executable to implement:

receiving a request from a process executing within one of a plurality of application servers for locked access to a primary state of session data comprising one or more attributes on a distributed store; and

granting locked access to the primary state to the process, wherein, while the primary state is locked for the process, other processes cannot access the primary state;

wherein said granting locked access to the primary state comprises sending a lock

token to the process, wherein only processes that have received a lock token can access the primary state.

32. (Currently amended) The computer-accessible storage medium as recited in claim 31, wherein the software instructions are further executable to implement requesting that the process release the locked access.

33. (Currently amended) The computer-accessible storage medium as recited in claim 31, wherein the software instructions are further executable to implement buffering one or more requests for locked access from one or more other processes executing within one or more of the plurality of application servers.

34. (Currently amended) The computer-accessible storage medium as recited in claim 33, wherein the software instructions are further executable to implement, if the process releases the locked access to the primary state, granting locked access to one of the other processes in response to the other process's buffered request.

35. (Currently amended) A computer-accessible storage medium, comprising software instructions executable to implement:

a process executing within one of a plurality of application servers requesting locked access to a primary state of session data stored by a distributed store;

the process receiving locked access, wherein, while the process holds the lock, other processes cannot access the primary state in the distributed store;

the process granting a thread-level lock to a portion of the primary state to a thread running within the process, wherein, while the thread holds the thread-level lock, other threads cannot access the portion of the primary state;

the process receiving a request to release the locked access; and

the process releasing the locked access in response to said request.

36. (Canceled)

37. (Currently amended) The computer-accessible storage medium as recited in claim 35, wherein the software instructions are further executable to implement the process releasing the lock when the process no longer requires locked access to the primary state.